Scalable Parallelization of the Sparse-Approximate-Inverse (SAI) Preconditioner for the Solution of Large-Scale Integral-Equation Problems†

Tahir Malas[1,2] and Levent Gürel[1,2]*
[1] Department of Electrical and Electronics Engineering
[2] Computational Electromagnetics Research Center (BiLCEM)
Bilkent University, TR-06800, Bilkent, Ankara, Turkey
E-mail: {tmalas,lgurel}@ee.bilkent.edu.tr

Introduction

Recently, there has been major progress in the development of parallel fast solvers for large-scale scattering problems [1, 2]. However, for complex targets and open-surface problems that should be solved with the electric-field integral equation (EFIE), fast solvers alone are not sufficient. Iterative solutions also require the development of robust and efficient preconditioners to obtain convergence in reasonable times.

In this paper, we consider efficient parallelization of the sparse approximate inverse (SAI) preconditioner in the context of the multilevel fast multipole algorithm (MLFMA). Then, we report the use of SAI in the solution of very large EFIE problems. The SAI preconditioner is important not only because it is a robust preconditioner that renders many difficult and large problems solvable, but also it can be utilized for the construction of more effective preconditioners [3, 4].

Load-Balancing of the SAI Preconditioner

The SAI preconditioner is computed with the Frobenius-norm-minimization technique [5]. Selection of the Frobenius norm enables the parallel computation of each row of SAI to be performed individually after some inter-process communication [6]. However, without an efficient load-balancing technique, generation phase of SAI can have inferior speedup, particularly for complex geometries.

We propose an efficient load-balancing method for the parallelization of SAI to obtain high scalability in the setup phase. The cost of the generation of a row of SAI is $\mathcal{O}(k^3)$, where k is the number of nonzero elements in that row. Note that this cost is different for the near-field generation phase, for which the cost of a row is proportional to the number of nonzero elements in that row. Hence, in a parallel implementation, if the near-field partitioning is also used for SAI, the SAI generation phase will be unbalanced.

The proposed load-balancing algorithm first determines generation cost of each row of SAI and then forms another partitioning of the near-field matrix for the parallel SAI setup. This SAI partitioning determines which processes generate which SAI rows. This way, we obtain a highly scalable setup phase of SAI. However, the application cost of the SAI preconditioner is in accordance with the original near-field partitioning. Therefore, the final distribution of SAI should be consistent with the near-field partitioning. Hence, the generated rows should be redistributed with inter-process communications.

The overhead of the redistribution of SAI rows can be eliminated by overlapping communications with computations. First, all processes initiate the receptions of the SAI rows that they would have with respect to the near-field partitioning but they do not generate, *i.e.,* they do not have with respect to the SAI partitioning. Then, all processes generate the rows in their SAI partitioning that do not belong to themselves with respect to the near-field partitioning and initiate the transfers of these rows. While the communications take place, local computations, *i.e.,* the generation of the rows that belong to a process with respect to both near-field and SAI partitioning, are performed. Finally, all

†This work was supported by the Scientific and Technical Research Council of Turkey (TUBITAK) under Research Grants 105E172 and 107E136, by the Turkish Academy of Sciences in the framework of the Young Scientist Award Program (LG/TUBA-GEBIP/2002-1-12), and by contracts from ASELSAN and SSM.

978-1-4244-3647-7/09/$25.00 ©2009 IEEE

for each row $i \in R_k^{near}$ **do**
  if row $i \notin R_k^{SAI}$ **then**
    $p =$ findProcId(i)
    start the reception of $\boldsymbol{m}_i$ from p ! Non-blocking communication
  endif
endfor
for each row $i \in R_k^{SAI}$ **do**
  if row $i \notin R_k^{near}$ **then**
    $p =$ findProcId(i)
    generate $\boldsymbol{m}_i$ and start the transfer to p ! Non-blocking communication
  endif
endfor
for each row $i \in R_k^{SAI}$ **do**
  if row $i \in R_k^{near}$ **then**
    generate $\boldsymbol{m}_i$
  endif
endfor
finish all non-blocking communications

Figure 1: Redistribution of the SAI rows according to the near-field partitioning. R_k^{near} and R_k^{SAI} denote the row indices of process k with respect to the near-field and SAI partitioning, respectively.

processes wait for the finalization of the non-blocking communications. We provide the algorithmic details of this approach in Fig. 1.

To show the effectiveness of the proposed approach, in Fig. 2, we show the SAI setup time of individual processes before and after the load balancing is applied. The problem is a complex target, called Flamme [7], which is solved with 32 processes. When the load-balancing method is not applied, significant load imbalance is observed, and this causes very low efficiency, particularly for large process numbers, as shown in Fig. 3. However, with the proposed load-balancing method, we obtain well-balanced setup times and superior speedup up to 128 processes.

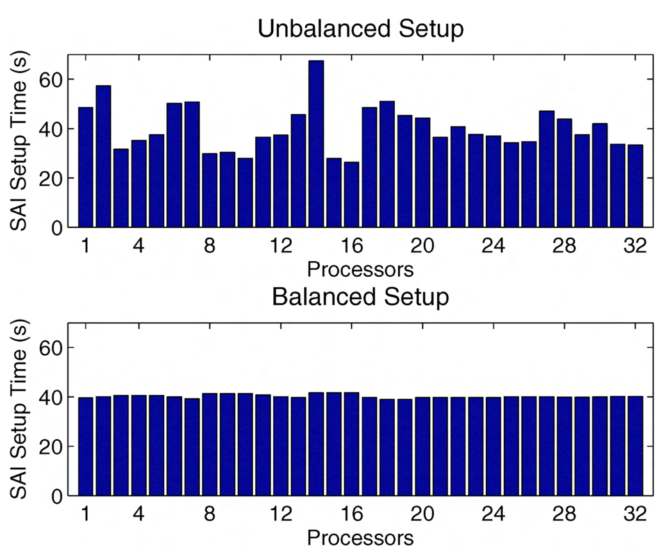

Figure 2: Unbalanced and balanced setup times of each processor in a 32-process job for the Flamme problem.

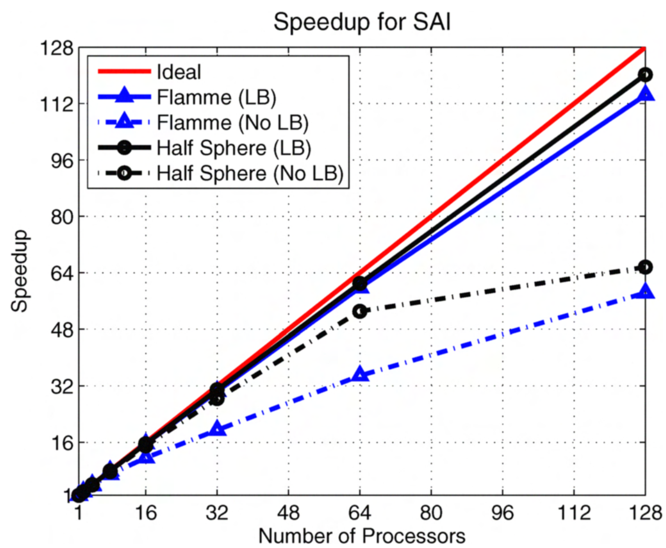

Figure 3: Speedup for SAI for the no-load-balance case (No LB) and after the load balancing is applied (LB).

Solution of Large EFIE Problems with SAI

By the help of the efficient SAI preconditioner, we are able to solve very large EFIE problems on moderate-cost parallel clusters. For example, in Table 1, we depict some of the problems that are solved with SAI on a 16-node cluster with two 3.0 GHz Intel Xeon quad-core processors and 16 GB of RAM per node. These problems include a patch with more than 33 million unknowns, a half-sphere with 15 million unknowns, and a reflector antenna with approximately 12 million unknowns. We note that these problems cannot be solved without a preconditioner or with simple preconditioners, such as diagonal or block-diagonal preconditioner. On the other hand, using an efficiently parallelized SAI preconditioner, we have been able to solve each problem in low iteration counts and moderate solution times. In particular, we obtain the solution of the patch problem involving 33 million unknowns in 7.5 hours; this is the largest open-surface problem solved with EFIE, to the best of our knowledge. We also note that the communication overhead is negligible compared to the SAI setup time, thanks to our efficient communication scheme.

Table 1: Solution of some very large EFIE problems.

Problem	Size (λ)	Number of MLFMA Levels	Number of Unknowns	SAI Comm. (secs)	SAI Setup (hours)	Number of Iters	Solution Time (hours)
Patch	288	12	32,999,808	8.6	1.5	70	7.8
Half Sphere	384	11	15,356,992	7.1	1.5	357	18.1
Reflector	214	11	11,967,620	6.6	0.5	336	12.7

Conclusion

For large open-surface problems that are modeled by EFIE, linear systems can be challenging to solve. Strong preconditioners with low computational complexity and parallel scalability need to be developed for such problems. In this work, we provide a parallel SAI preconditioner that satisfies these requirements. Thanks to the efficiently parallelized SAI, we have been able to solve very large ill-conditioned EFIE problems in a moderate-cost parallel cluster. We note that SAI preconditioner not only renders solution of large open-surface problems, it also speeds up convergence of complex closed-surface problems that can make use of the CFIE.

References

[1] S. Velamparambil and W. C. Chew, "Analysis and performance of a distributed memory multilevel fast multipole algorithm," *IEEE Trans. Antennas Propagat.*, vol. 53, no. 8, pp. 2719–2727, Aug. 2005.

[2] L. Gürel and Ö. Ergül, "Fast and accurate solutions of integral-equation formulations discretised with tens of millions of unknowns," *Electronics Lett.*, vol. 43, pp. 499–500, 2007.

[3] T. Malas, Ö. Ergül, and L. Gürel, "Sequential and parallel preconditioners for large-scale integral-equation problems," in *2007 Computational Electromagnetics Workshop, İzmir, Turkey*, August 2007, pp. 35–43.

[4] P.-L. Rui, R.-S. Chen, D.-X. Wang, and E.-N. Yung, "Spectral two-step preconditioning of multilevel fast multipole algorithm for the fast monostatic RCS calculation," *IEEE Trans. Antennas Propagat.*, vol. 55, no. 8, pp. 2268–2275, Aug. 2007.

[5] M. Benzi and M. Tuma, "A comparative study of sparse approximate inverse preconditioners," *Appl. Numer. Math.*, vol. 30, no. 2–3, pp. 305–340, 1999.

[6] T. Malas and L. Gürel, "Accelerating the multilevel fast multipole algorithm with the sparse-approximate-inverse (SAI) preconditioning," *SIAM J. Sci. Comput.*, Dec. 2008, accepted for publication.

[7] L. Gürel, H. Bağcı, J.-C. Castelli, A. Cheraly, and F. Tardivel, "Validation through comparison: Measurement and calculation of the bistatic radar cross section of a stealth target," *Radio Science*, vol. 38, no. 3, pp. 1046–1058, 2003.